Computer-Aided VLSI System Design
Lab3: Synthesis Lab: Design Compiler

TA: 朱怡蓁 r10943012@ntu.edu.tw

Introduction

In this lab, you will learn:

1. Basic concept about synthesis
2. How to use Synopsys Design Compiler (in text mode)

Environmental Setup

1. Make sure you can run design compiler and VCS.

```
source /usr/cad/synopsys/CIC/synthesis.cshrc
source /usr/cad/synopsys/cshrc
```

2. Make a working directory

```
mkdir Lab3
```

```
cd Lab3
```

Copy Files from NTU Cool

1. Upload all the files downloaded from NTU Cool to your working directory (Lab3.zip)
2. Check if you have these files

Files/Folder	Description
`Lab3_test_alu.v`	The testbench of the design file (ALU)
`Lab3_alu.v`	The source design file (ALU)

Synopsys Design Compiler

1. Check the search path and library is set as the following:

```
set company "CIC"
set designer "Student"
set search_path      {CAD_path/CBDK013_TSMC_Artisan/CIC/SynopsysDC/db  \

/home/raid7_2/course/cvsd/CBDK_IC_Contest/CIC/SynopsysDC/lib  \

/home/raid7_2/course/cvsd/CBDK_IC_Contest/CIC/SynopsysDC/db  \
                     $search_path }
set link_library     "typical.db slow.db fast.db dw_foundation.sldb"
set target_library   "typical.db slow.db fast.db"
set symbol_library   "generic.sdb"
set synthetic_library "dw_foundation.sldb"
```

2. The function of Lab3_alu.v is from your previous Lab. We will try to use this sample and practice Synopsys synthesis tool step by step. We will show the text-mode with the strings in the text blocks.

3. Check the RTL simulation.

```
vcs -full64 -R Lab3_test_alu.v Lab3_alu.v
```

4. Build your working directory by yourself, copy all files and start up text mode.

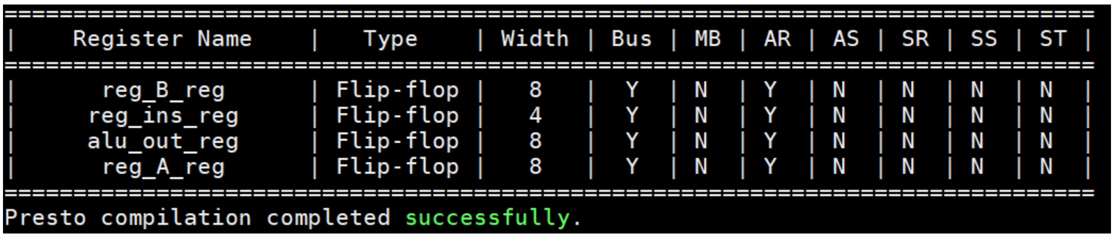
```
dc_shell
```

3. Load file with the following command:

```
read_file -format verilog {./Lab3_alu.v}
```

```
dc_shell> read_file -format verilog {./Lab3_alu.v}
```

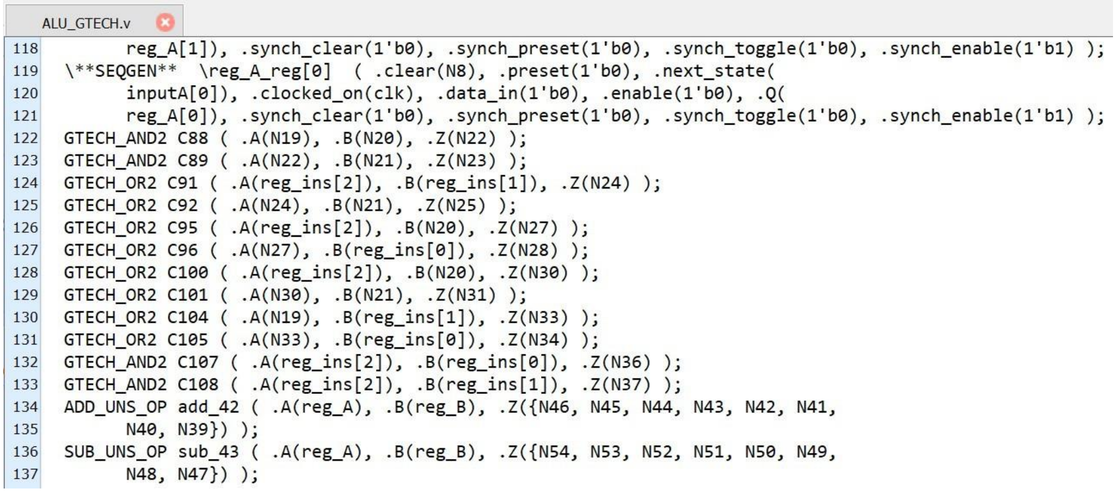
4. Check the log information. If any error or warning message, you have to fix it! After that, checking all the registers are filp-flop type. You have to modify your verilog code, if there is the latch in your circuit!



```
==============================================================================
|    Register Name     |    Type    | Width | Bus | MB | AR | AS | SR | SS | ST |
==============================================================================
|      reg_B_reg       | Flip-flop |   8   |  Y  | N  | Y  | N  | N  | N  | N  |
|     reg_ins_reg      | Flip-flop |   4   |  Y  | N  | Y  | N  | N  | N  | N  |
|     alu_out_reg      | Flip-flop |   8   |  Y  | N  | Y  | N  | N  | N  | N  |
|      reg_A_reg       | Flip-flop |   8   |  Y  | N  | Y  | N  | N  | N  | N  |
==============================================================================
Presto compilation completed successfully.
```

5. Write out the current design and check if each macro is mapped as you expect.

```
write –format verilog -hierarchy -output ALU_GTECH.v
```



```
       reg_A[1]), .synch_clear(1'b0), .synch_preset(1'b0), .synch_toggle(1'b0), .synch_enable(1'b1) );
  \**SEQGEN**  \reg_A_reg[0]  ( .clear(N8), .preset(1'b0), .next_state(
       inputA[0]), .clocked_on(clk), .data_in(1'b0), .enable(1'b0), .Q(
       reg_A[0]), .synch_clear(1'b0), .synch_preset(1'b0), .synch_toggle(1'b0), .synch_enable(1'b1) );
  GTECH_AND2 C88 ( .A(N19), .B(N20), .Z(N22) );
  GTECH_AND2 C89 ( .A(N22), .B(N21), .Z(N23) );
  GTECH_OR2 C91 ( .A(reg_ins[2]), .B(reg_ins[1]), .Z(N24) );
  GTECH_OR2 C92 ( .A(N24), .B(N21), .Z(N25) );
  GTECH_OR2 C95 ( .A(reg_ins[2]), .B(N20), .Z(N27) );
  GTECH_OR2 C96 ( .A(N27), .B(reg_ins[0]), .Z(N28) );
  GTECH_OR2 C100 ( .A(reg_ins[2]), .B(N20), .Z(N30) );
  GTECH_OR2 C101 ( .A(N30), .B(N21), .Z(N31) );
  GTECH_OR2 C104 ( .A(N19), .B(reg_ins[1]), .Z(N33) );
  GTECH_OR2 C105 ( .A(N33), .B(reg_ins[0]), .Z(N34) );
  GTECH_AND2 C107 ( .A(reg_ins[2]), .B(reg_ins[0]), .Z(N36) );
  GTECH_AND2 C108 ( .A(reg_ins[2]), .B(reg_ins[1]), .Z(N37) );
  ADD_UNS_OP add_42 ( .A(reg_A), .B(reg_B), .Z({N46, N45, N44, N43, N42, N41,
       N40, N39}) );
  SUB_UNS_OP sub_43 ( .A(reg_A), .B(reg_B), .Z({N54, N53, N52, N51, N50, N49,
       N48, N47}) );
```

How many "GTECH_OR2" are there after HDL translation? _____

6. Specify the clock as period 10ns. (100 MHz). We also set “**don’t touch network**” and “**fixhold**” attributes.

```
create_clock -name "clk" -period 10 -waveform {"0" "5"} {"clk"}
set_dont_touch_network [find clock clk]
set_fix_hold clk
```

7. And then type in following command to change the wire load model:

```
set_operating_conditions "typical" -library "typical"
set_wire_load_model -name "ForQA" -library "typical"
set_wire_load_mode "segmented"
```

8. Set operating environment, including input delay and output delay attributes

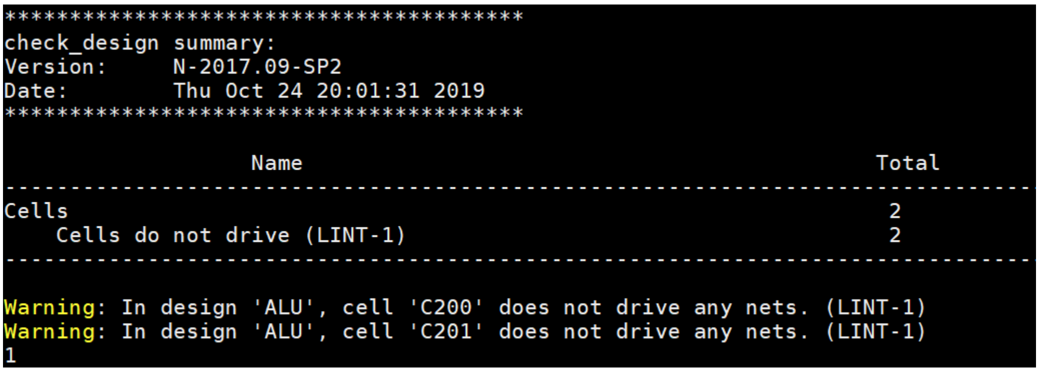
```
set_input_delay -clock clk 2.5 inputA[*]
set_input_delay -clock clk 3.8 inputB[*]
set_input_delay -clock clk 4.5 instruction[*]
set_input_delay -clock clk 5.2 reset
set_output_delay -clock clk 8 alu_out[*]
```

9. Set design constraints, including max area, max fanout and max transition.

```
set_boundary_optimization "*"
set_fix_multiple_port_nets –all -buffer_constant
set_max_area 0
set_max_fanout 8 ALU
set_max_transition 1 ALU
```

10. Checks the current design for consistency.

```
check_design
```

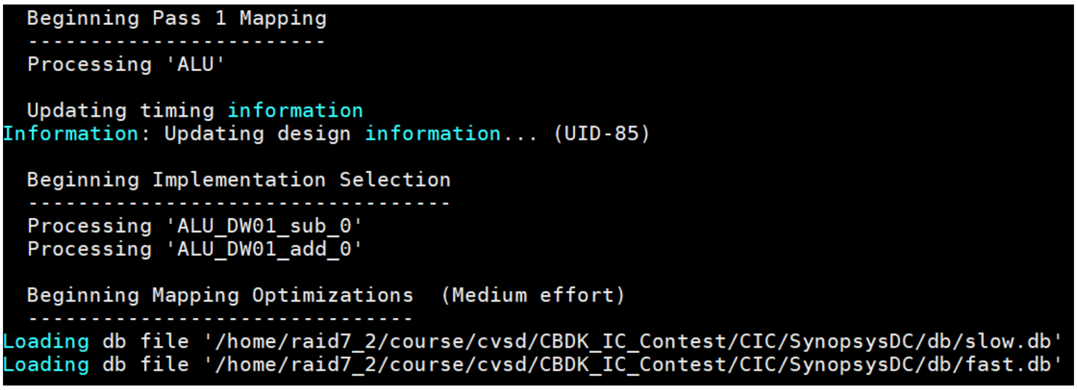


```
****************************************
check_design summary:
Version:     N-2017.09-SP2
Date:        Thu Oct 24 20:01:31 2019
****************************************

                   Name                                            Total
--------------------------------------------------------------------------------
Cells                                                               2
    Cells do not drive (LINT-1)                                     2
--------------------------------------------------------------------------------

Warning: In design 'ALU', cell 'C200' does not drive any nets. (LINT-1)
Warning: In design 'ALU', cell 'C201' does not drive any nets. (LINT-1)
1
```

10. Start to perform optimization of ALU

```
compile -map_effort medium
```

The mapping details will be displayed on the console.

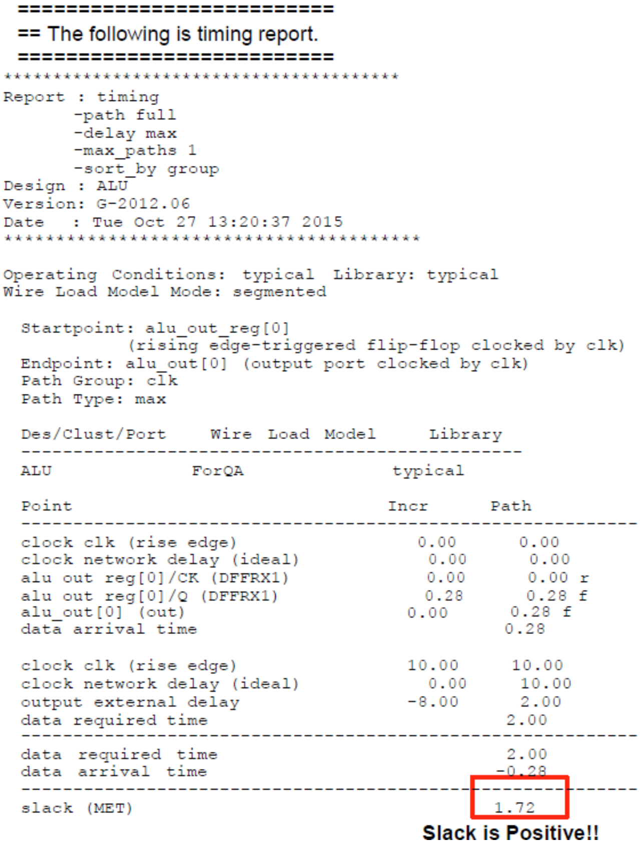


```
  Beginning Pass 1 Mapping
  ------------------------
  Processing 'ALU'

  Updating timing information
Information: Updating design information... (UID-85)

  Beginning Implementation Selection
  ----------------------------------
  Processing 'ALU_DW01_sub_0'
  Processing 'ALU_DW01_add_0'

  Beginning Mapping Optimizations  (Medium effort)
  -------------------------------
Loading db file '/home/raid7_2/course/cvsd/CBDK_IC_Contest/CIC/SynopsysDC/db/slow.db'
Loading db file '/home/raid7_2/course/cvsd/CBDK_IC_Contest/CIC/SynopsysDC/db/fast.db'
```

11. Few seconds later, we will get our gate level circuit. We must to check our circuit met our conditions or not at first. And we can report timing with following command. and to generate the *ALU.timing* file to record the timing information of optimized design.

```
report_timing -path full -delay max -max_paths 1 -nworst 1 > ALU.timing
```

Check the slack is positive (meet the timing constraint) or negative.

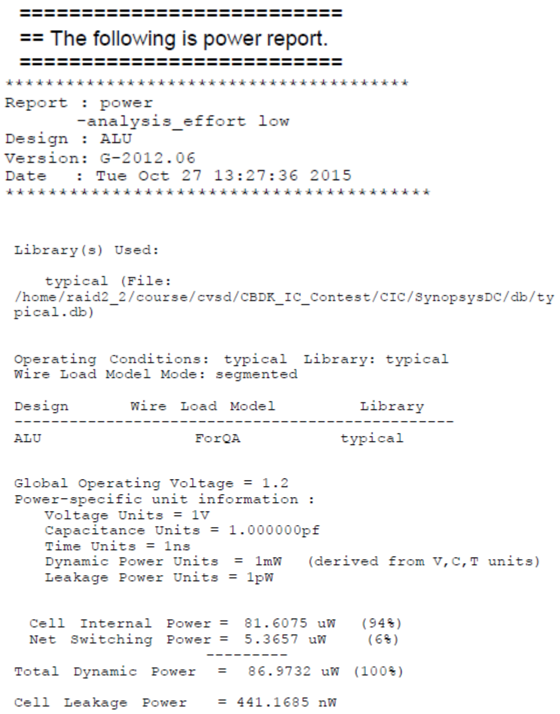


```
 ==========================
 == The following is timing report.
 ==========================
****************************************
Report : timing
        -path full
        -delay max
        -max_paths 1
        -sort_by group
Design : ALU
Version: G-2012.06
Date   : Tue Oct 27 13:20:37 2015
****************************************

Operating Conditions: typical  Library: typical
Wire Load Model Mode: segmented

  Startpoint: alu_out_reg[0]
              (rising edge-triggered flip-flop clocked by clk)
  Endpoint: alu_out[0] (output port clocked by clk)
  Path Group: clk
  Path Type: max

  Des/Clust/Port     Wire Load Model       Library
  ------------------------------------------------
  ALU                ForQA                 typical

  Point                                    Incr       Path
  -----------------------------------------------------------
  clock clk (rise edge)                    0.00       0.00
  clock network delay (ideal)              0.00       0.00
  alu out reg[0]/CK (DFFRX1)               0.00       0.00 r
  alu out reg[0]/Q (DFFRX1)                0.28       0.28 f
  alu_out[0] (out)                         0.00       0.28 f
  data arrival time                                   0.28

  clock clk (rise edge)                   10.00      10.00
  clock network delay (ideal)              0.00      10.00
  output external delay                   -8.00       2.00
  data required time                                  2.00
  -----------------------------------------------------------
  data required time                                  2.00
  data arrival time                                  -0.28
  -----------------------------------------------------------
  slack (MET)                                         1.72
```

Slack is Positive!!

12. We can also report power with the following command, and to generate the ALU.power file to record the power consumption of optimized design.

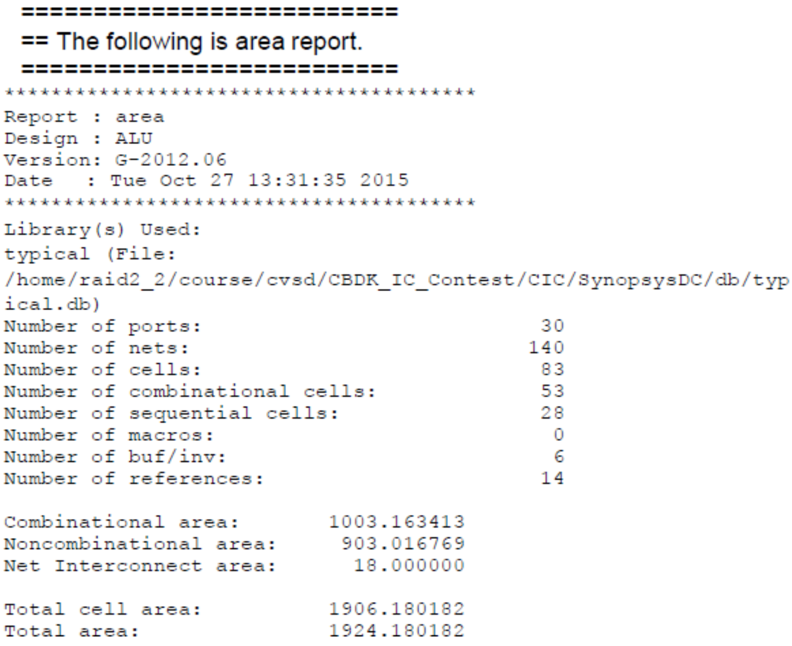
```
report_power > ALU.power
```



```
===========================
== The following is power report.
===========================
****************************************
Report : power
        -analysis_effort low
Design : ALU
Version: G-2012.06
Date   : Tue Oct 27 13:27:36 2015
****************************************


 Library(s) Used:

    typical (File:
 /home/raid2_2/course/cvsd/CBDK_IC_Contest/CIC/SynopsysDC/db/ty
 pical.db)


 Operating Conditions: typical  Library: typical
 Wire Load Model Mode: segmented

 Design       Wire Load Model         Library
 ------------------------------------------------
 ALU                 ForQA          typical


 Global Operating Voltage = 1.2
 Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000pf
    Time Units = 1ns
    Dynamic Power Units  = 1mW   (derived from V,C,T units)
    Leakage Power Units = 1pW


   Cell  Internal  Power =  81.6075 uW   (94%)
   Net  Switching  Power =  5.3657 uW     (6%)
                          ---------
 Total  Dynamic  Power   =  86.9732 uW  (100%)

 Cell  Leakage  Power    = 441.1685 nW
```

13. We can also report area with the following command, and to generate the ALU.area file to record the area of optimized design.

```
report_area -nosplit > ALU.area
```



```
===========================
== The following is area report.
===========================
****************************************
Report : area
Design : ALU
Version: G-2012.06
Date   : Tue Oct 27 13:31:35 2015
****************************************
Library(s) Used:
typical (File:
/home/raid2_2/course/cvsd/CBDK_IC_Contest/CIC/SynopsysDC/db/typ
ical.db)
Number of ports:                            30
Number of nets:                            140
Number of cells:                            83
Number of combinational cells:              53
Number of sequential cells:                 28
Number of macros:                            0
Number of buf/inv:                           6
Number of references:                       14

Combinational area:        1003.163413
Noncombinational area:      903.016769
Net Interconnect area:       18.000000

Total cell area:           1906.180182
Total area:                1924.180182
```

14. If the result is met your requirement, Synthesis is ending. Then, we must export the design to a file. It will save the all the settings and results in **ALU.ddc**).

```
write -hierarchy -format ddc
```

15. We can also generate a file to store all design constraints we’ve set.

```
write_sdc ALU.sdc
```

16. Finally, to save the timing information, you have to type the following command in the command line. That will generate the timing information of this design.

```
write_sdf -version 2.1 ALU.sdf
```

17. You should also write gate-level netlist for gate-level simulation.

```
write -format verilog -hierarchy -output ALU_syn.v
```

18. For verilog gate-level simulation, you may add

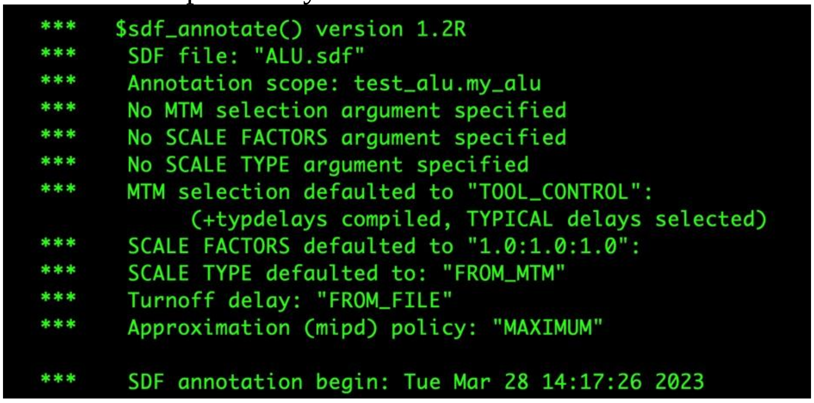
```
$sdf_annotate(“ALU.sdf”, my_alu);
```

in initial block in your test bench to use timing information for simulation.

19. Run the gate-level simulation in the command line

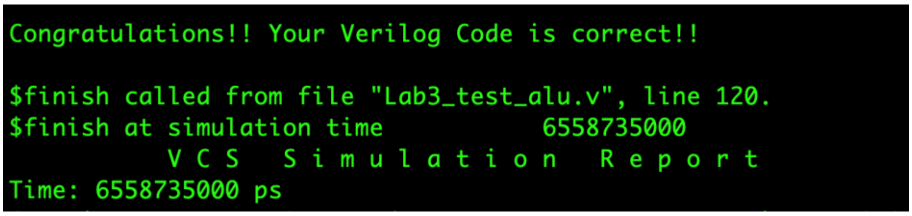
```
vcs -full64 -R Lab3_test_alu.v ALU_syn.v -v +neg_tchk
/home/raid7_2/course/cvsd/CBDK_IC_Contest/CIC/Verilog/tsmc13_neg.v
```

Show that sdf path delays are annotated.



```
   ***    $sdf_annotate() version 1.2R
   ***     SDF file: "ALU.sdf"
   ***     Annotation scope: test_alu.my_alu
   ***     No MTM selection argument specified
   ***     No SCALE FACTORS argument specified
   ***     No SCALE TYPE argument specified
   ***     MTM selection defaulted to "TOOL_CONTROL":
                (+typdelays compiled, TYPICAL delays selected)
   ***     SCALE FACTORS defaulted to "1.0:1.0:1.0":
   ***     SCALE TYPE defaulted to: "FROM_MTM"
   ***     Turnoff delay: "FROM_FILE"
   ***     Approximation (mipd) policy: "MAXIMUM"

   ***     SDF annotation begin: Tue Mar 28 14:17:26 2023
```

Show your gate-level simulation results.



```
Congratulations!! Your Verilog Code is correct!!

$finish called from file "Lab3_test_alu.v", line 120.
$finish at simulation time           6558735000
           V C S   S i m u l a t i o n   R e p o r t
Time: 6558735000 ps
```

Checkpoints

Please check with TAs before leaving this lab to make sure the following goals are accomplished and to get credits.

1. Answer the question: How many “GTECH_OR2” are there after HDL translation? _____
2. Snapshots “sdf path delays is successfully annotated”. (The snapshot must contain your account name

3. Snapshots “your gate-level simulation results”. (The snapshot must contain your account name)

Submission

1. **Due Tuesday, Oct. 17th, 19:00**
2. Answer the question and take the snapshot of the results shown in previous section and submit to NTU COOL in pdf format.